(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 September 2001 (27.09.2001)

PCT

(10) International Publication Number
WO 01/72075 A1

(51) International Patent Classification[7]: H04Q 7/38, H04L 9/12

(21) International Application Number: PCT/EP01/02646

(22) International Filing Date: 9 March 2001 (09.03.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
0006668.8 21 March 2000 (21.03.2000) GB

(71) Applicant *(for all designated States except US)*: TELEFONAKTIEBOLAGET LM ERICSSON (publ) [SE/SE]; S-126 25 Stockholm (SE).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: SILEN, Lars [FI/FI]; Mankholmsvagen 3, FIN-2380 Esbo (FI).

(74) Agent: LIND, Robert; Marks & Clerk, 4220 Nash Court, Oxford Business Park South, Oxford, Oxfordshire OX4 2RU (GB).

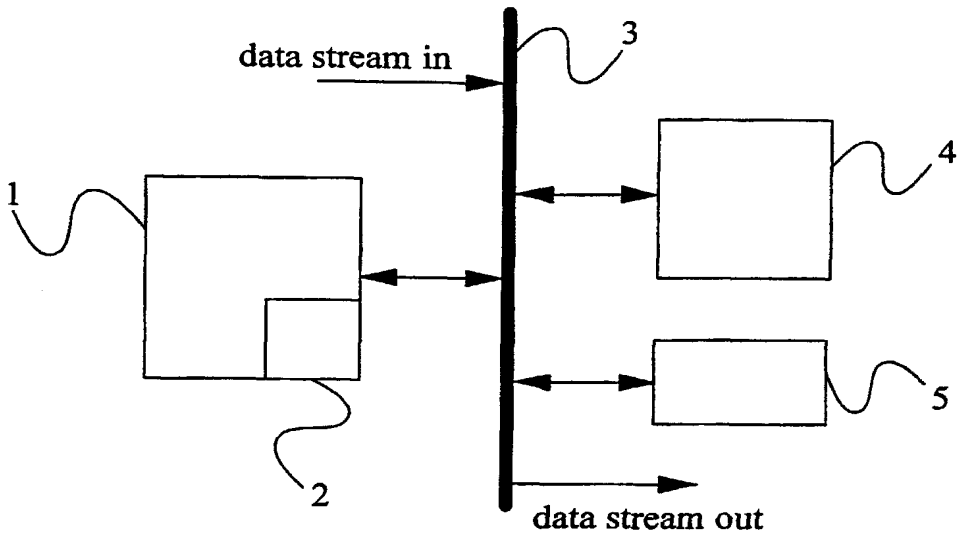
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: ENCRYPTING AND DECRYPTING



(57) Abstract: A method of encrypting/decrypting a sequence of data blocks by performing an XOR operation between the data blocks and respective crypto masks. The method comprises, prior to or upon receipt of a first of said data blocks: initialising a crypto engine (1,2); using the initialised crypto engine (1,2) to generate a sequence of crypto masks corresponding to respective data blocks; storing the crypto masks in a buffer (5); and subsequently following receipt of each data block, performing an XOR operation between the data block and the corresponding crypto mask to encrypt/decrypt the data block.

WO 01/72075 A1

ENCRYPTING AND DECRYPTING

Field of the Invention

The present invention relates to the encrypting and decrypting of data and in particular, though not necessarily, to the encrypting and decrypting of data in a telecommunications system.

Background to the invention

In many communications systems there is a need for the encryption of transmitted data in order to ensure the security of the systems. This is particularly true in telecommunications systems such as mobile telecommunications networks where data may be sent over an open interface to which third parties may listen. One common encryption/decryption method involves the generation of a cryptographic mask (or "crypto mask") using an encryption key which is known to both ends of the communication channel. A block of data is encrypted or decrypted by performing an XOR operation between the data and the mask. Generally, for each block of data in a data stream, a new crypto mask is generated and used. (The 3GPP paper TS 25.301 V3.2.0 (1999-10) describes this process for third generation mobile networks.)

The encryption and subsequent decryption of data streams can place high demands on the responsible processors. In the so-called third generation mobile telecommunications systems this is particularly so as the processors, both in the networks and in the user terminals, may be required to handle multiple parallel data streams. Whilst increases in processor power, involving *inter alia* the use of hardware assistance, help to mitigate the problem, this is generally being outstripped by the increase in data processing requirements.

Modern microprocessors generally use external memories that have an access time up to 30 times slower than the speed of the processor. The relative slowness of the external memory is dealt with through the use of a fast intermediate memory called "cache". As

long as the code or data required by the processor is in the cache memory (having previously been transferred there from the external memory), accesses run essentially at the processor speed without the need for any slow wait states. When data or program code is not found in the cache, the processor has to break the flow of computations to update the cache from the external memory. The cache update may require hundreds of clock cycles (corresponding to hundreds of lost machine instructions).

When many simultaneous data streams are handled, the responsible processor has to do a large number of context switches to switch between different data streams, operating system handling etc. In most cases a context switch also means a cache re-load and thus the loss of a large number of processor instruction cycles. In the case of encryption and decryption using a crypto mask, it is also necessary to generate a new crypto mask for each successive block of data in a data stream. This places a significant load on the processor given that the encryption/decryption engine (or "crypto engine", which may be implemented in software, on an ASIC, or on a dedicated encrypting/decrypting processor) is initialised prior to each mask generation step and following receipt of each new data block (initialisation of the crypto engine requires much more time than does the actual generation of the crypto mask with which the data is XORed). The effect of the context switches due to multiple data stream processing and to a large number of crypto engine initialisations results in a large loss of processor capacity.

Summary of the Present Invention.

The inventor of the present invention has recognised that, as the generation of a crypto mask is independent of the actual data to be encrypted/decrypted, a number of crypto masks can be generated and buffered, upon or prior to receipt of a first block of a sequence of data blocks. Thus, for the sequence, only a single initialisation of the crypto engine is required *vis-à-vis* mask generation.

According to a first aspect of the present invention there is provided a method of encrypting/decrypting a sequence of data blocks by performing an XOR operation

between the data blocks and respective crypto masks, the method comprising, prior to or upon receipt of a first of said data blocks:

initialising a crypto engine;

using the initialised crypto engine to generate a sequence of crypto masks corresponding to respective data blocks;

storing the crypto masks in a buffer; and subsequently

following receipt of each data block, performing an XOR operation between the data block and the corresponding crypto mask to encrypt/decrypt the data block.

It will be appreciated that by generating all of the crypto masks required to encrypt/decrypt a sequence of data blocks prior to or upon receipt of a first of the blocks, only a single initialisation of the crypto engine is required. This is in contrast to the prior art where the engine is initialised, a first mask generated, the XOR operation performed, and the process repeated for each subsequent block (i.e. each data block is treated as a separate entity). Thus, embodiments of the present invention result in a significant reduction in the load on the responsible processor for a given data throughput.

It will be further appreciated that said step of initialising the crypto engine typically involves the loading of instructions relating to the crypto engine from an external memory into a memory cache of the responsible processor.

The present invention is applicable in particular to telecommunication systems and more particularly to mobile telecommunications systems such as the Universal Mobile Telecommunications System (UMTS). For example, the invention may be employed to encrypt/decrypt data sent over the air interface of a mobile telecommunications network, where encryption and decryption is performed both on the network side and on the subscriber side. The invention is also applicable to other communication systems as well as to systems arranged to encrypt/decrypt data for the purpose of secure storage.

According to a second aspect of the present invention there is provided apparatus for encrypting/decrypting a sequence of data blocks by performing an XOR operation between the data blocks and respective crypto masks, the apparatus comprising:

a processor:

an external memory storing instructions for causing the processor to operate as a crypto engine;

a cache arranged to receive said instructions from the external memory during initialisation of the crypto engine by the processor; and

a buffer,

the processor being arranged in use to initialise the crypto engine prior to or upon receipt of a first of said data blocks, to generate a sequence of crypto masks corresponding to respective data blocks, and to store the masks in said buffer, whereupon, following receipt of each data block, the processor can perform an XOR operation between the data block and the corresponding crypto mask to encrypt/decrypt the data block.

A mobile telecommunications terminal comprising the apparatus of the above second aspect of the present invention.

A node of a telecommunications network comprising the apparatus of the above second aspect of the present invention.

Brief Description of the Drawings

Figure 1 illustrates schematically an encryption/decryption system embodying the present invention and which is implemented in software;

Figure 2 illustrates schematically a crypto engine embodying the present invention and which is implemented in hardware;

Figure 3 illustrates an encryption/decryption system comprising the hardware crypto engine of Figure 2;

Figure 4 is a flow diagram illustrating the operation of the systems of Figures 1 and 3;

Figure 5 is a flow diagram illustrating the freeing of buffer space in the systems of Figures 1 and 3;

Figure 6 is a flow diagram illustrating the handling of missing data blocks in a sequence of blocks;

Figure 7 illustrates schematically the operation of the systems of Figures 1 and 3 where data blocks are received in sequence; and

Figure 8 illustrates schematically the operation of the systems of Figures 1 and 3 where data blocks are received out of sequence.

Detailed Description of Certain Embodiments

There is illustrated in Figure 1, in very general terms, an encryption/decryption system. A microprocessor 1 has an internal cache memory 2 and is connected to a data bus 3. Also connected to the data bus 3 are a ROM memory 4 and a RAM buffer 5. Assuming that the system is operating in an encryption mode, a data stream comprising a sequence of data blocks is placed on the data bus 3 via an input port (not shown). The encrypted blocks are similarly placed on the bus 3 and are passed to an output port (not shown). In a decryption mode, encrypted blocks are placed on the data bus 3 via the input port and the decrypted blocks, also placed on the bus 3, are output via the output port. By way of example, the received data blocks may correspond to the payloads of IP datagrams.

The memory 4 is a relatively slow memory and stores instruction code and data for operating the microprocessor 1. In particular, the memory 4 stores code for providing a crypto engine (see the 3GPP paper TS 25.301 V3.2.0 (1999-10)), the code being passed to the microprocessor 1 for temporary storage in the cache 2 during an initialisation phase of the engine. The initialisation phase continues with the generation of crypto masks which are passed by the microprocessor 1 to the buffer 5 for temporary storage therein.

Figure 2 illustrates a hardware implementation of a crypto engine which may be used in place of the software engine of the system of Figure 1. The engine may be implemented

using an Application Specific Integrated Circuit (ASIC) or a Field Programmable Gate Array (FPGA) and comprises a number of registers (start block, end block, ... , crypto key) which are required in order to generate a crypto mask. The registers are set by the processor during initialisation of the crypto engine. The register information is then used by the engine to generate the required sequence of crypto masks which are stored directly into the buffer without intervention from the microprocessor.

Figure 3 illustrates the interworking between the hardware crypto engine of Figure 2 and a processor in an encryption/decryption system, where it is assumed that the incoming data stream is received over an ATM transport network.

Figure 4 is a flow diagram illustrating the method of operation of the systems of Figures 1 and 3. When a new data stream is opened and the first block of that stream received, the processor obtains the appropriate cryptographic key, the count (which represents an index to the crypto mask buffer), the identity of the stream (or bearer), the direction (i.e. encrypt or decrypt), and determines the number of blocks N for which crypto masks are to be created and the length of these blocks. These parameters represent the inputs to the crypto engine required in order to generate the first crypto mask. Generated crypto masks are fed back to provide an input to the crypto engine to generate subsequent masks. As crypto masks are independent of the data to be encrypted/decrypted, a number of crypto masks can be pre-calculated.

The processor creates a temporary small buffer for storing the incoming data blocks, and a second crypto buffer for storing the generated crypto masks as well as the encrypted/decrypted blocks. After the required set of crypto masks has been generated and stored in the crypto buffer, the processor sets up an array containing pointers to the start of each successive mask in the crypto buffer, and flags to indicate if respective blocks have been encrypted/decrypted. The first block is then handled by XORing it with the first block mask stored in the buffer. The resulting block is written into the crypto buffer on top of the corresponding mask and the corresponding flag set to indicate that the first block has been processed.

For each block of the stream which is subsequently received, the processor determines whether or not a mask exists in the crypto buffer for that block. If a mask does exist, then the block is XORed with the mask, the result written on top of the mask in the crypto buffer, and the corresponding flag set to indicate that the block has been processed. The result is immediately available for forwarding to the output. A counter is used to record the number of handled data blocks written into the crypto buffer. When the counter reaches a value of N (i.e. the current crypto mask has been consumed), a new buffer and corresponding crypto mask have to be created. Figure 5 is a flow diagram illustrating the process which is carried out following the handling of the final (Nth) block in a sequence in order to forward the encrypted/decrypted sequence and free the crypto buffer space.

In the event that a received block has a block number which is higher than the highest number for which a mask exists in the buffer, then if the block number falls within the next expected set of N blocks a new buffer is created for those next N blocks and the received block stored in that buffer. The crypto engine is reinitialised and the next set of N crypto masks generated. The received block is then XORed with the appropriate mask. In the event that the block number of a received block is higher than the next buffer space (due for example to data blocks being sent via different paths or to a network error), an error state is generated and, for example, the block discarded.

The process for handling missing data blocks is illustrated in the flow diagram of Figure 6. More particularly, a Timeout counter is used to ensure that the system does not wait indefinitely for a missing block and to flush and free the data buffer in case of errors. The Timeout counter is reset to zero when a block is correctly received, and subsequently counts up with every operating system "tick". If the counter reaches some predefined (large) value, this indicates that an error has occurred and that the buffer should be flushed (all processed blocks are forwarded and the buffer marked as free).

Figure 7 illustrates schematically the process of encrypting/decrypting a sequence of data blocks forming a data stream, where crypto masks are pre-calculated for a set of N blocks. Figure 7 assumes that the blocks are received in the correct sequence. Figure 8

illustrates an alternative scenario, where a series of data blocks are received in the incorrect order. It will be noted that the use of the block pointer array flag {Blk#1, Blk#2, Blk#3, ...} allows blocks to be marked as handled or not handled. There is thus no need for the blocks to be received (and handled) in order.

5

It will be appreciated by the person of skill in the art that various modifications may be made to the above described embodiments without departing from the scope of the present invention.

10

Claims

1. A method of encrypting/decrypting a sequence of data blocks by performing an XOR operation between the data blocks and respective crypto masks, the method comprising, prior to or upon receipt of a first of said data blocks:

initialising a crypto engine;

using the initialised crypto engine to generate a sequence of crypto masks corresponding to respective data blocks;

storing the crypto masks in a buffer; and subsequently

following receipt of each data block, performing an XOR operation between the data block and the corresponding crypto mask to encrypt/decrypt the data block.

2. A method according to claim 1, wherein said step of initialising the crypto engine involves the loading of instructions relating to the crypto engine from an external memory into a memory cache of the responsible processor.

3. A method according to claim 1 or 2, wherein said sequence of data blocks comprise user data in a telecommunication systems.

4. A method according to claim 3, wherein said telecommunications system is a mobile telecommunications system.

5. A method according to claim 4, wherein said received sequence of data blocks comprises user data received over, or to be sent over, the air interface.

6. Apparatus for encrypting/decrypting a sequence of data blocks by performing an XOR operation between the data blocks and respective crypto masks, the apparatus comprising:

a processor:

an external memory storing instructions for causing the processor to operate as a crypto engine; and

a buffer,

the processor being arranged in use to initialise the crypto engine prior to or upon receipt of a first of said data blocks, to generate a sequence of crypto masks corresponding to respective data blocks, and to store the masks in said buffer, whereupon, following receipt of each data block, the processor can perform an XOR operation between the data block and the corresponding crypto mask to encrypt/decrypt the data block.

7. Apparatus according to claim 6 and comprising a cache arranged to receive said instructions from the external memory during initialisation of the crypto engine by the processor.

8. Apparatus for encrypting/decrypting a sequence of data blocks by performing an XOR operation between the data blocks and respective crypto masks, the apparatus comprising:

a processor:

an Application Specific Integrated Circuit (ASIC) or a Field Programmable Gate Array (FPGA) arranged to operate as a crypto engine; and

a buffer,

the processor being arranged in use to initialise the crypto engine prior to or upon receipt of a first of said data blocks, to generate a sequence of crypto masks corresponding to respective data blocks, and to store the masks in said buffer, whereupon, following receipt of each data block, the processor can perform an XOR operation between the data block and the corresponding crypto mask to encrypt/decrypt the data block.

9. A mobile telecommunications terminal comprising the apparatus of any one of claims 6 to 8.

10. A node of a telecommunications network comprising the apparatus of any one of claims 6 to 8.

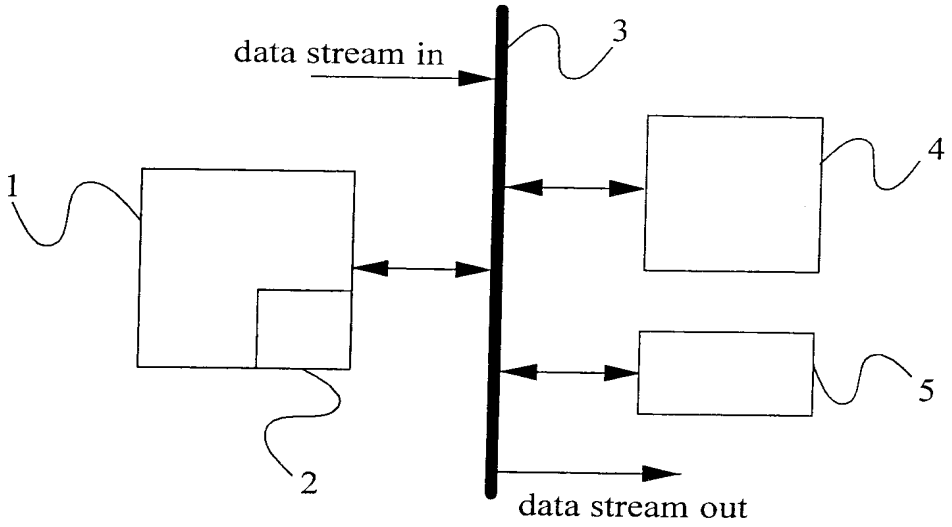


Figure 1

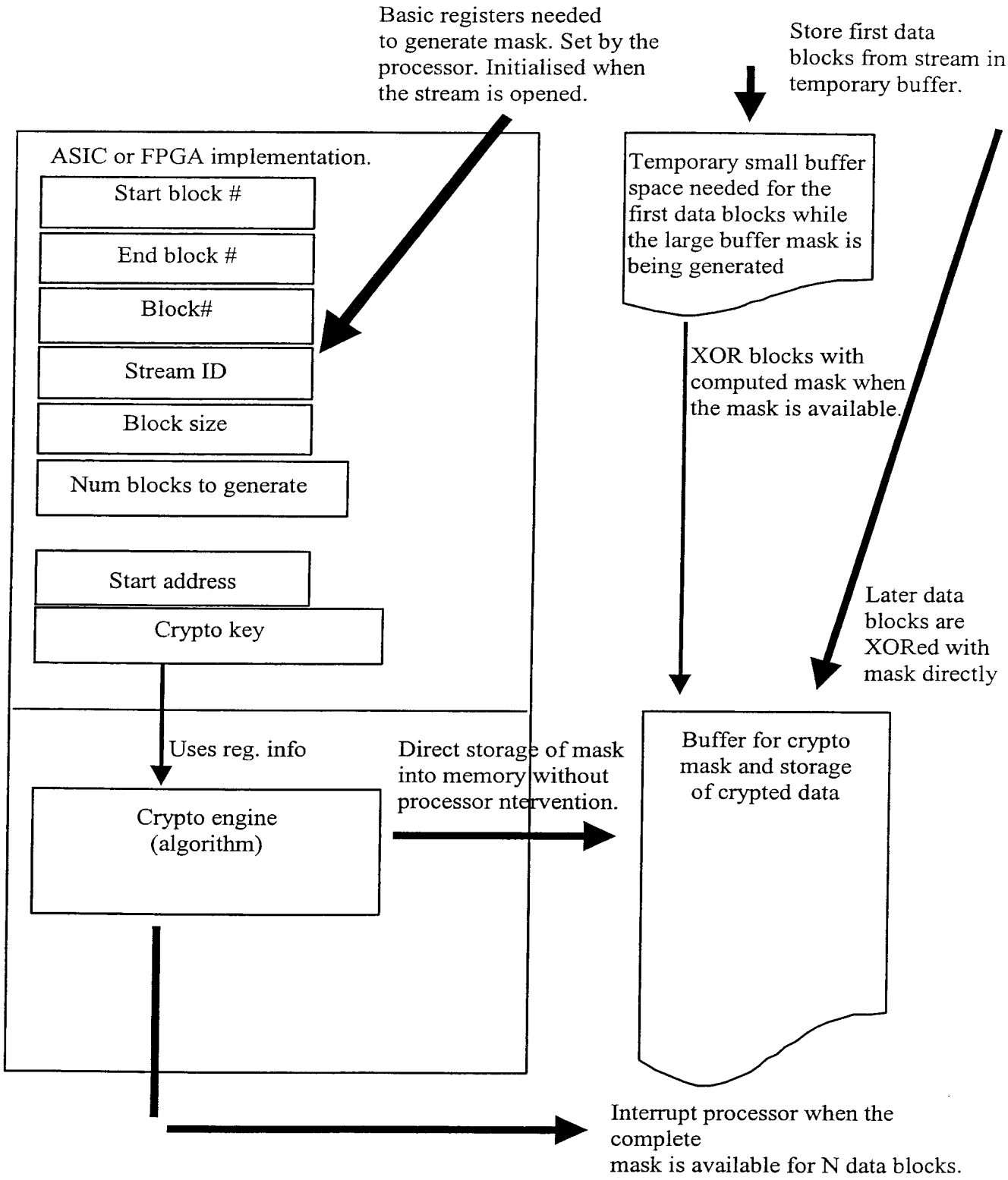

Basic registers needed to generate mask. Set by the processor. Initialised when the stream is opened.
Store first data blocks from stream in temporary buffer.
ASIC or FPGA implementation.
Start block #
End block #
Block#
Stream ID
Block size
Num blocks to generate
Start address
Crypto key
Temporary small buffer space needed for the first data blocks while the large buffer mask is being generated
XOR blocks with computed mask when the mask is available.
Later data blocks are XORed with mask directly
Uses reg. info
Crypto engine (algorithm)
Direct storage of mask into memory without processor ntervention.
Buffer for crypto mask and storage of crypted data
Interrupt processor when the complete mask is available for N data blocks.

Figure 2

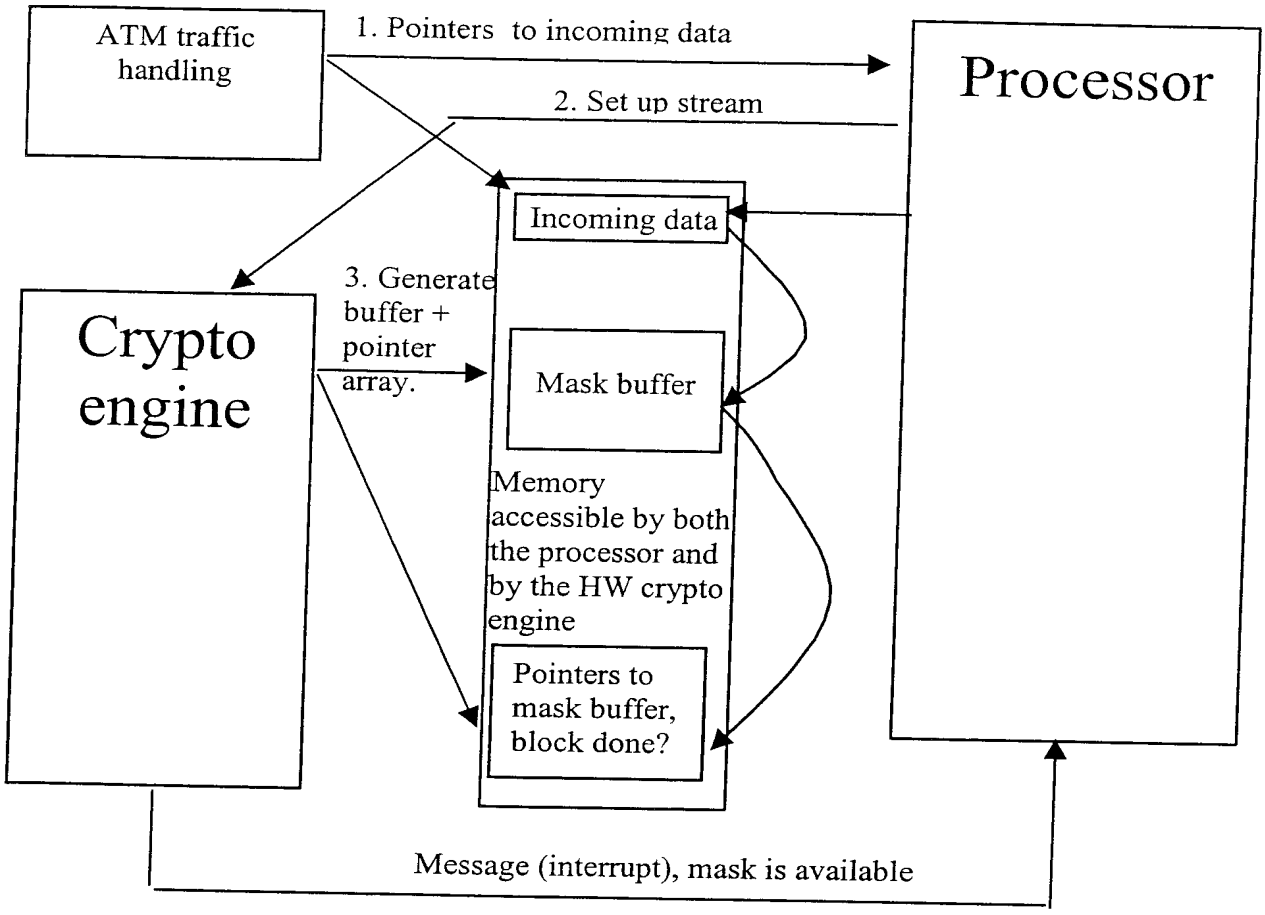

ATM traffic handling
1. Pointers to incoming data
2. Set up stream
Processor
Incoming data
3. Generate buffer + pointer array.
Crypto engine
Mask buffer
Memory accessible by both the processor and by the HW crypto engine
Pointers to mask buffer, block done?
Message (interrupt), mask is available

Figure 3

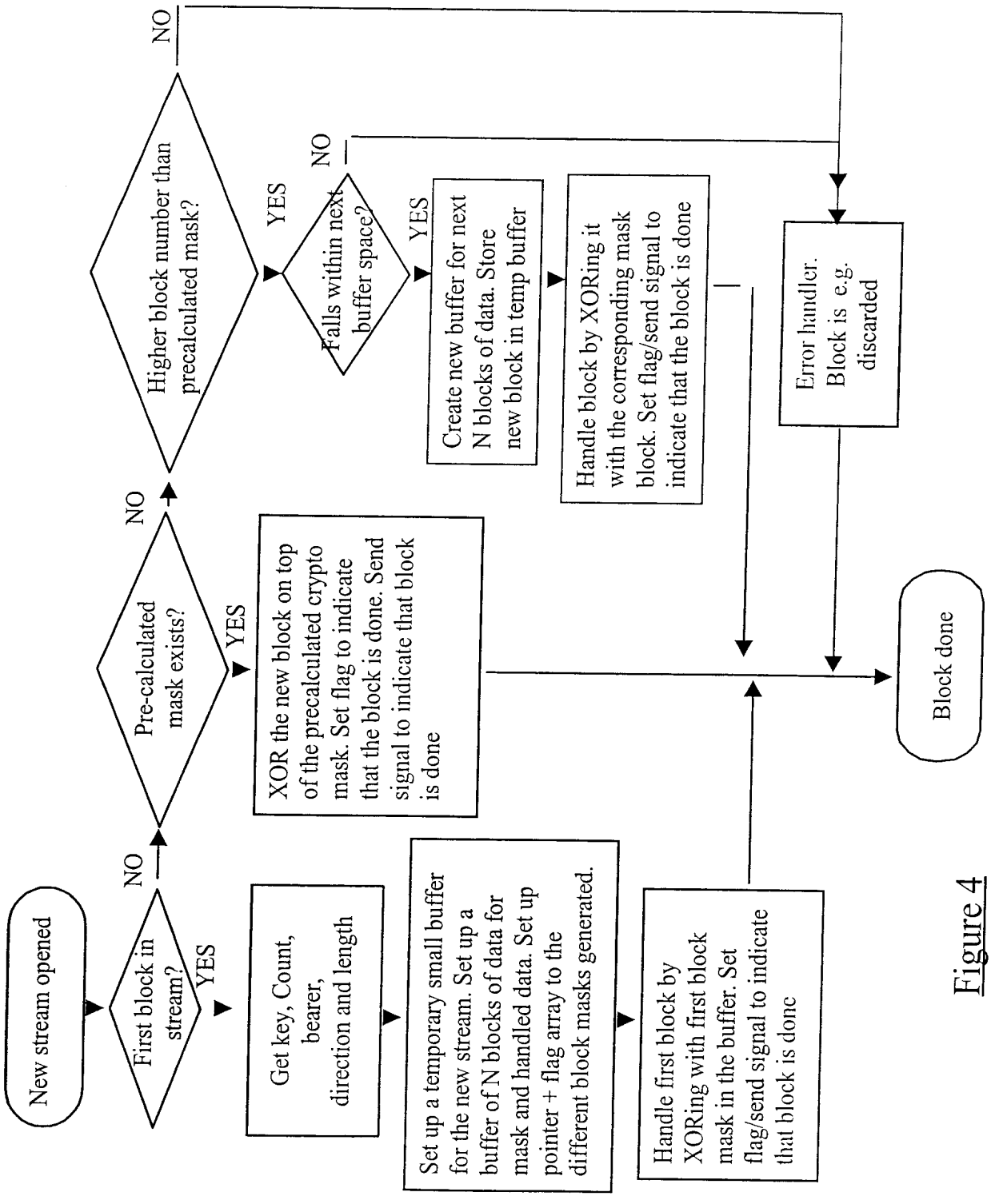

New stream opened
First block in stream?
YES
NO
Get key, Count, bearer, direction and length
Pre-calculated mask exists?
YES
NO
XOR the new block on top of the precalculated crypto mask. Set flag to indicate that the block is done. Send signal to indicate that block is done
Set up a temporary small buffer for the new stream. Set up a buffer of N blocks of data for mask and handled data. Set up pointer + flag array to the different block masks generated.
Handle first block by XORing with first block mask in the buffer. Set flag/send signal to indicate that block is done
Higher block number than precalculated mask?
YES
NO
Falls within next buffer space?
YES
NO
Create new buffer for next N blocks of data. Store new block in temp buffer
Handle block by XORing it with the corresponding mask block. Set flag/send signal to indicate that the block is done
Error handler. Block is e.g. discarded
Block done

Figure 4

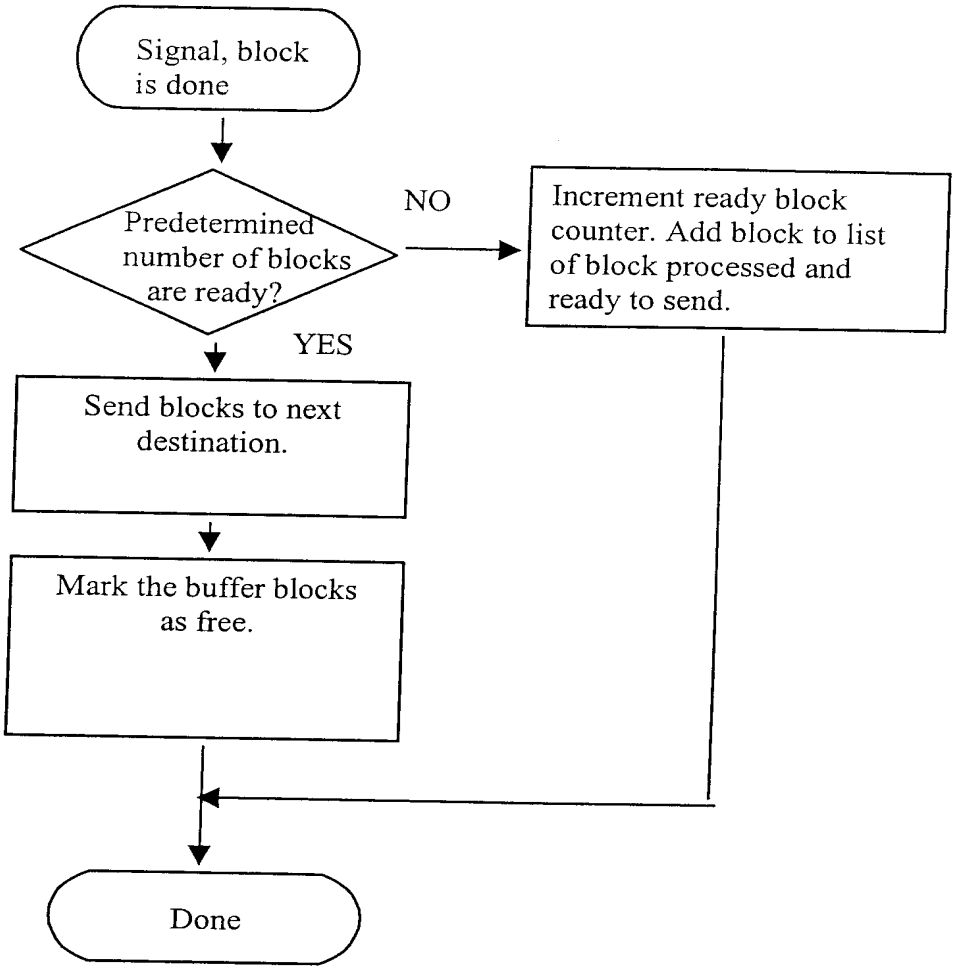

Signal, block is done
Predetermined number of blocks are ready?
NO
Increment ready block counter. Add block to list of block processed and ready to send.
YES
Send blocks to next destination.
Mark the buffer blocks as free.
Done

Figure 5

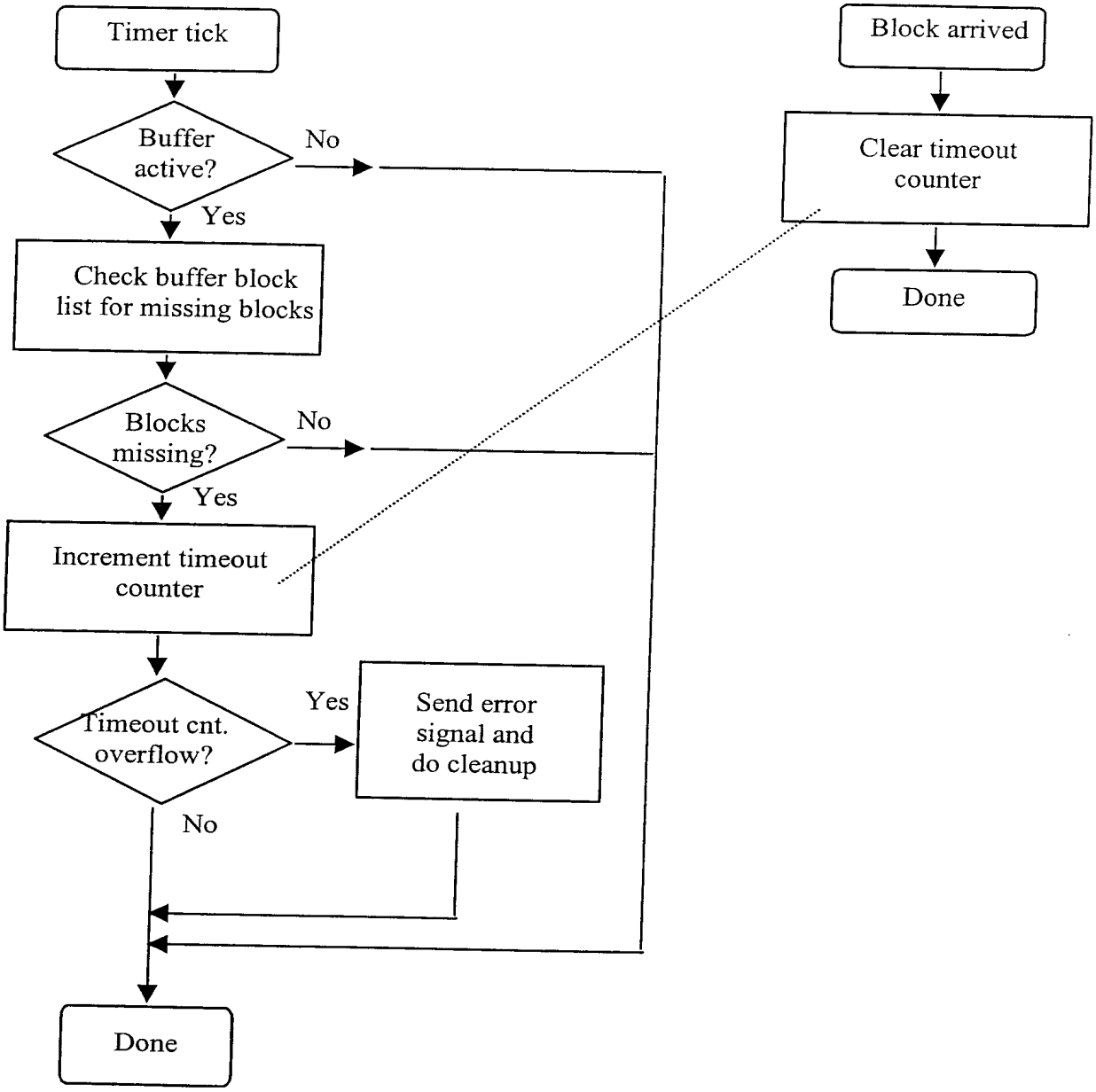

Timer tick
Buffer active?
No
Yes
Check buffer block list for missing blocks
Blocks missing?
No
Yes
Increment timeout counter
Timeout cnt. overflow?
Yes
Send error signal and do cleanup
No
Done
Block arrived
Clear timeout counter
Done

Figure 6

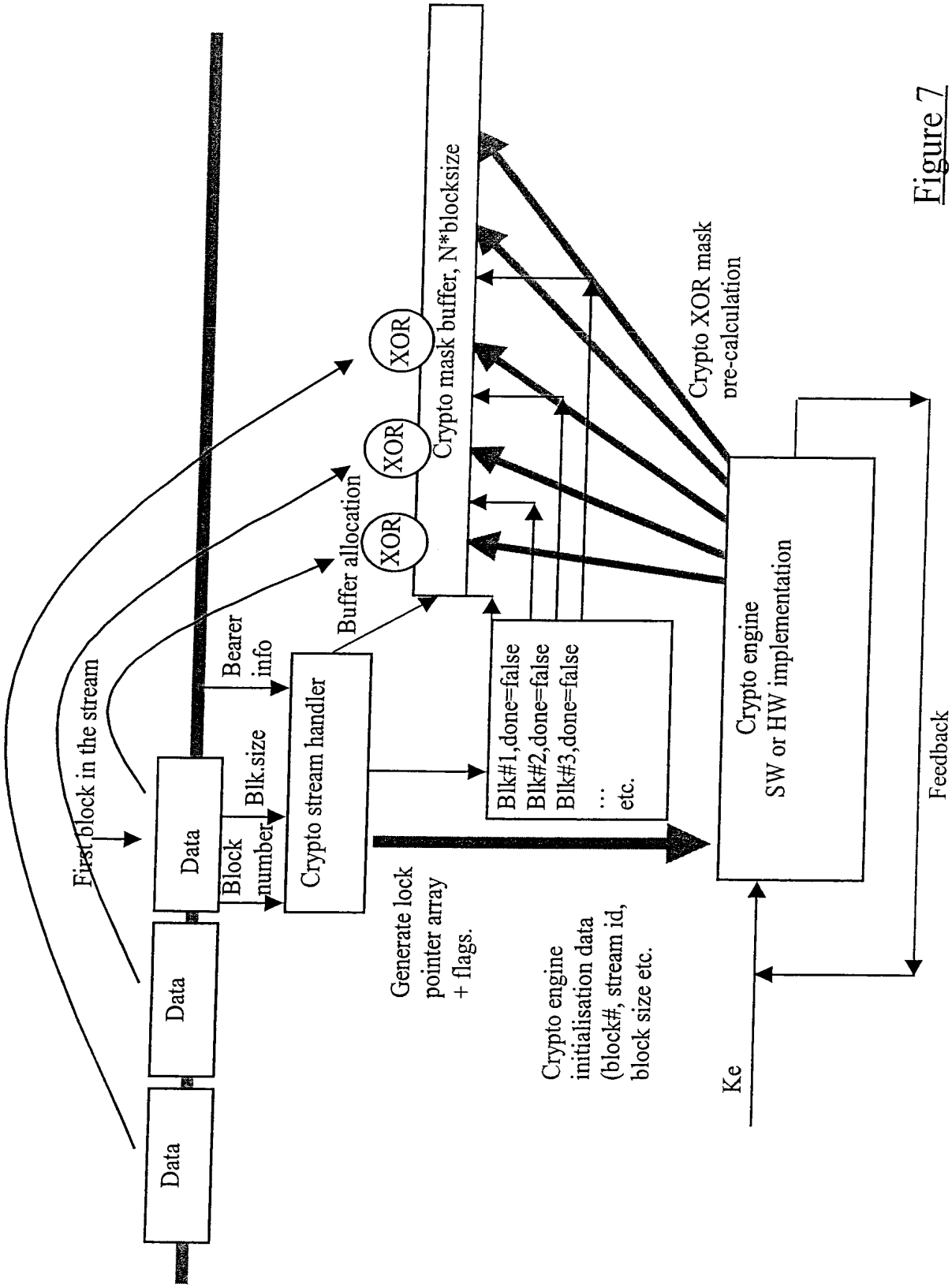

Data
Data
Data
First block in the stream
Block number
Blk.size
Bearer info
Crypto stream handler
Buffer allocation
XOR
XOR
XOR
Crypto mask buffer, N*blocksize
Blk#1,done=false
Blk#2,done=false
Blk#3,done=false
...
etc.
Crypto XOR mask pre-calculation
Crypto engine
SW or HW implementation
Feedback
Generate lock pointer array + flags.
Crypto engine initialisation data (block#, stream id, block size etc.
Ke

Figure 7

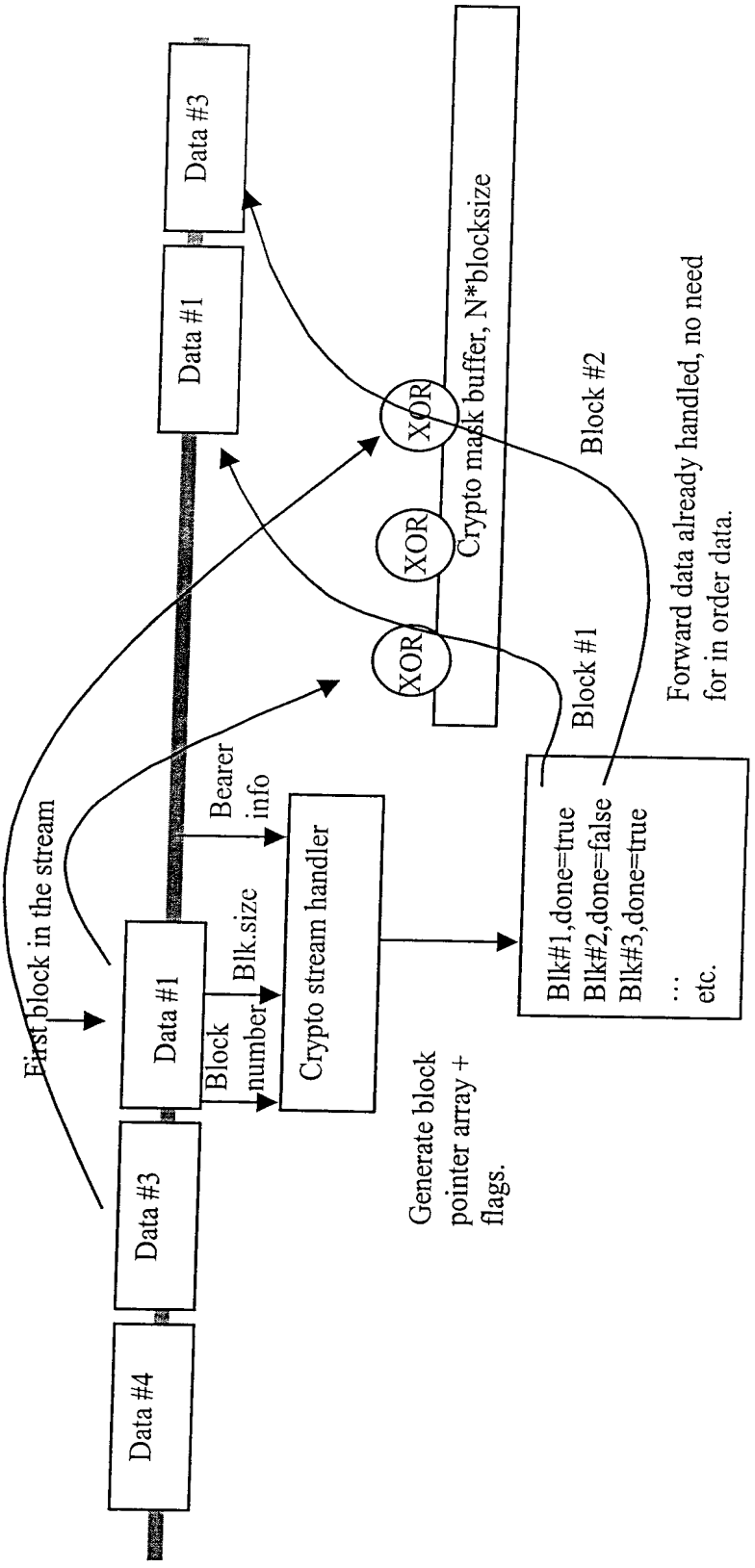

Data #4
Data #3
Data #1
Data #3
First block in the stream
Data #1
Block number
Blk.size
Bearer info
Crypto stream handler
Generate block pointer array + flags.
XOR
XOR
XOR
Crypto mask buffer, N*blocksize
Blk#1,done=true
Blk#2,done=false
Blk#3,done=true
…
etc.
Block #1
Block #2
Forward data already handled, no need for in order data.

Figure 8

INTERNATIONAL SEARCH REPORT

In ational Application No
PCT/EP 01/02646

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04Q7/38 H04L9/12

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04Q H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, WPI Data, PAJ, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 366 288 A (IBM) 2 May 1990 (1990-05-02) page 3, line 3-9 page 4, line 7-19	1,2,6-8
A	US 5 444 781 A (LYNN KERRY E ET AL) 22 August 1995 (1995-08-22) column 2, line 60 -column 3, line 68	1,2,6-8

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
8 August 2001	16/08/2001

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Weinmiller, J

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

In ational Application No
PCT/EP 01/02646

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0366288 A	02-05-1990	US 5016275 A	14-05-1991
		JP 3080645 A	05-04-1991
US 5444781 A	22-08-1995	US 5345508 A	06-09-1994
		AU 7602394 A	21-03-1995
		WO 9506373 A	02-03-1995